Lecture Topics

- Today: Combinational Circuits (H&H 2.1-2.9)
- Next: Sequential Circuits (H&H 3.1-3.5)

Announcements

- Self-study Module #1 (this week)
- Project #1 (due no later than 5/17)
- Project #2 (due no later than 5/19)
- Reminder: use Pi array
Computer Project #1

- Due Wednesday, 5/17 (by 11:59 PM)
- Focuses on computing environment:
  - UNIX tutorial
  - Using the "vim" editor
  - Using the "handin" system

Computer Project #2

- Due Friday, 5/19 (by 11:59 PM)
- Focuses on design of eight functions:
  - Present() – entries 0 or 1
  - a() thru g() – entries 0, 1 or X
Combinational Circuits

- Circuit design based on Boolean algebra
- Three equivalent representations
  - algebraic expressions
  - truth tables
  - circuit diagram

Example: Exclusive OR

- Expression in Boolean algebra:
  \[ F(A,B) = A'B + AB' \]
- Truth table:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>F(A,B)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
- Circuit diagram:

- Canonical Sum-of-Products Form:
  \[ F(A,B) = A'B + AB' \]

- The expression is the sum of a series of products, where each product is a minterm.

- A minterm is a product where each variable is present (complemented or uncomplemented).
Minterms

- For a function with two inputs, there are four possible minterms:
  
  m0: $A'B'$
  m1: $A'B$
  m2: $AB'$
  m3: $AB$

- Canonical SOP form has a subset of all possible minterms

Minterms

- For a function with three inputs, there are eight possible minterms:
  
  m0: $A'B'C'$
  m1: $A'B'C$
  m2: $A'BC'$
  m3: $A'BC$
  m4: $AB'C'$
  m5: $AB'C$
  m6: $ABC'$
  m7: $ABC$

- For a function with four inputs, there are sixteen possible minterms
Examples

The following are in canonical SOP form:

- \( F(A,B) = A'B' + A'B + AB' \)
- \( G(A,B,C) = AB'C + ABC' + ABC \)
- \( H(A,B,C,D) = A'B'C'D' + A'B'CD + ABCD' \)

Alternate notation (minterm lists)

The following are equivalent:

- \( F(A,B) = A'B' + A'B + AB' \)
- \( F(A,B) = m0 + m1 + m2 \)
- \( F(A,B) = \text{minterms}(0, 1, 2) \)
Additional Examples:

- $F(A,B,C) = AB'C + ABC' + ABC$
  
  $= minterms(5, 6, 7)$

- $F(A,B,C,D) = A'B'C'D' + A'B'CD + ABCD'$
  
  $= minterms(0, 3, 14)$

Truth tables and minterms

- Canonical Sum-of-Products Form:
  
  $F(A,B) = A'B + AB' = minterms(1, 2)$

- Truth table:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>F(A,B)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Canonical Sum-of-Products form makes it easy to convert between representations:

Given: \( F(A,B,C) = \text{minterms}( 5, 6, 7 ) \)

- \( F(A,B,C) = AB'C + ABC' + ABC \)
- truth table has 1’s in rows m5, m6, m7
- circuit diagram has three AND gates (one for each minterm) and one OR gate

Minimization

Ideally, a Boolean expression will be as simple as possible and still generate the correct values

Note that there are an infinite number of Boolean expressions that represent the same function:

\[
F(A,B) = A'B + AB' \\
= A'B + AB' + AB' \\
= A'B + AB' + AB' + AB'
\]
Minimization Criteria

The minimized (optimal, simplified) Boolean expression is the one which has:

- the fewest number of gates
- the fewest number of inputs to gates

Reminder: we’re working with the complete gate set { NOT, AND, OR }

Minimization Techniques

Three techniques:

- Algebraic manipulation
- Karnaugh map
- Quine-McCluskey algorithm
Apply the postulates and theorems of Boolean algebra:

\[ AB' + AB = A(B' + B) \quad \text{(distributive law)} \]
\[ = A(1) \quad \text{(complement law)} \]
\[ = A \quad \text{(identity law)} \]

\[ F(A,B,C) = A'BC' + A'BC + ABC' + ABC \]
\[ = A'B(C' + C) + ABC' + ABC \]
\[ = A'B(1) + ABC' + ABC \]
\[ = A'B + ABC' + ABC \]
\[ = A'B + AB(C' + C) \]
\[ = A'B + AB(1) \]
\[ = A'B + AB \]
\[ = A'B + AB \]
\[ = (A' + A)B \]
\[ = (1)B \]
\[ = B \]
Karnaugh Map

Fill in the entries in a K-map, then inspect it to identify the optimal expression

Karnaugh map is rectangular and has one entry for each minterm – adjacent minterms can be combined (same rules as algebraic manipulation)

K-map for function with 2 inputs

<table>
<thead>
<tr>
<th></th>
<th>A'</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>B'</td>
<td>m0</td>
<td>m2</td>
</tr>
<tr>
<td>B</td>
<td>m1</td>
<td>m3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>m0</td>
<td>m2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>m1</td>
<td>m3</td>
<td></td>
</tr>
</tbody>
</table>
Example: $F(A,B) = AB' + AB$

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>A'</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B'</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Minimized function: $F(A,B) = A$

Example: $F(A,B) = AB' + A'B$

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>A'</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B'</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Minimized function: $F(A,B) = AB' + A'B$
K-map for function with 3 inputs

<table>
<thead>
<tr>
<th></th>
<th>A'B'</th>
<th>A'B</th>
<th>AB</th>
<th>AB'</th>
</tr>
</thead>
<tbody>
<tr>
<td>C'</td>
<td>m0</td>
<td>m2</td>
<td>m6</td>
<td>m4</td>
</tr>
<tr>
<td>C</td>
<td>m1</td>
<td>m3</td>
<td>m7</td>
<td>m5</td>
</tr>
</tbody>
</table>

K-map for function with 3 inputs

<table>
<thead>
<tr>
<th>AB</th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>m0</td>
<td>m2</td>
<td>m6</td>
<td>m4</td>
</tr>
<tr>
<td>1</td>
<td>m1</td>
<td>m3</td>
<td>m7</td>
<td>m5</td>
</tr>
</tbody>
</table>
**Ex: Algebraic Manipulation**

\[ F(A,B,C) = \text{minterms}(2, 3, 6, 7) \]

\[ = A'BC' + A'BC + ABC' + ABC \]

\[ = A'B(C' + C) + ABC' + ABC \]

\[ = A'B + ABC' + ABC \]

\[ = A'B + AB(C' + C) \]

\[ = A'B + AB \]

\[ = B \]

---

**Ex: Karnaugh Map**

<table>
<thead>
<tr>
<th></th>
<th>A'B'</th>
<th>A'B</th>
<th>AB</th>
<th>AB'</th>
</tr>
</thead>
<tbody>
<tr>
<td>C'</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ F(A,B,C) = B \]
Ex: Karnaugh Map

<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

F(A,B,C) = B

Application: Majority Function

The majority function is true when more than half of the inputs are true.

Majority function on 3 inputs:
\[ F(A,B,C) = \text{minterms} (3, 5, 6, 7) \]

\[ = A'BC + AB'C + ABC' + ABC \]

Gates: 8

Inputs: 19

---

\[ F(A,B,C) = \text{minterms} (3, 5, 6, 7) \]

\[ = A'BC + AB'C + ABC' + ABC \]

\[ = A'BC + AB'C + ABC' + ABC + ABC + ABC \]

\[ = (A'+A)BC + AC(B'+B) + AB(C'+C) \]

\[ = BC + AC + AB \]

\[ = AB + AC + BC \]
F(A, B, C) = minterms\((3, 5, 6, 7)\)

Minimized function: $F(A, B, C) = AB + AC + BC$
Minimized function: \( F(A, B, C) = AB + AC + BC \)

K-map for function with 4 inputs

<table>
<thead>
<tr>
<th></th>
<th>A'B'</th>
<th>A'B</th>
<th>AB</th>
<th>AB'</th>
</tr>
</thead>
<tbody>
<tr>
<td>C'D'</td>
<td>m0</td>
<td>m4</td>
<td>m12</td>
<td>m8</td>
</tr>
<tr>
<td>C'D</td>
<td>m1</td>
<td>m5</td>
<td>m13</td>
<td>m9</td>
</tr>
<tr>
<td>CD</td>
<td>m3</td>
<td>m7</td>
<td>m15</td>
<td>m11</td>
</tr>
<tr>
<td>CD'</td>
<td>m2</td>
<td>m6</td>
<td>m14</td>
<td>m10</td>
</tr>
</tbody>
</table>
K-map for function with 4 inputs

<table>
<thead>
<tr>
<th></th>
<th>AB</th>
</tr>
</thead>
<tbody>
<tr>
<td>CD</td>
<td>00</td>
</tr>
<tr>
<td>00</td>
<td>m0</td>
</tr>
<tr>
<td>01</td>
<td>m1</td>
</tr>
<tr>
<td>11</td>
<td>m3</td>
</tr>
<tr>
<td>10</td>
<td>m2</td>
</tr>
</tbody>
</table>

Ex: \( F(A,B,C,D) = \text{minterms}(2, 5, 8, 9, 10, 11, 12) \)

\[
F(A,B,C,D) = \text{minterms}(2, 5, 8, 9, 10, 11, 12)
\]

\[
F(A,B,C,D) = AB' + AC'D' + B'CD' + A'BC'D
\]
Ex: \( F(A,B,C,D) = \text{minterms}( 3,4,5,7,9,13,14,15 ) \)

<table>
<thead>
<tr>
<th>( AB )</th>
<th>( CD )</th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>01</td>
<td></td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ F(A,B,C,D) = BD + A'BC' + AC'D + ABC + A'CD \]

Ex: \( F(A,B,C,D) = \text{minterms}( 3,4,5,7,9,13,14,15 ) \)

<table>
<thead>
<tr>
<th>( AB )</th>
<th>( CD )</th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>01</td>
<td></td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ F(A,B,C,D) = A'BC' + AC'D + ABC + A'CD \]
Order is important!

- Start with 1’s which are isolated
- Find 1’s that can only be included in 2-cover
- Find 1’s that can only be included in 4-cover
- Find 1’s that can only be included in 8-cover
- Continue until all 1’s covered at least once

K-maps are "circular":

\[ F = B C D + \overline{B} D + \overline{A} B \]
Irrelevant minterms (don’t cares)

- For some functions, there are particular input combinations which cannot occur or particular output values that won’t be used.
- Those minterms are irrelevant (we don’t care whether the value of that minterm is a 0 or a 1).
- Useful during minimization.

Consider three functions (D,E,F) which use the same three inputs (A,B,C):

If A or C is true, then function D is true.
If A or B is true, then function E is true.
Function F is true if exactly one of the inputs is true. Furthermore, the value of function F is irrelevant when functions D and E are true.
If A or C is true, then function D is true

If A or B is true, then function E is true

Function F is true if exactly one of the inputs is true. Furthermore, the value of function F is irrelevant when functions D and E are true

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

45

If A or C is true, then function D is true

If A or B is true, then function E is true

Function F is true if exactly one of the inputs is true. Furthermore, the value of function F is irrelevant when functions D and E are true

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>X</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>X</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>X</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>X</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>X</td>
</tr>
</tbody>
</table>

46
K-map for $D(A,B,C)$

\[
\begin{array}{cccc}
A & B & C & 00 & 01 & 11 & 10 \\
\hline
0 & 0 & 0 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 \\
\end{array}
\]

$D(A,B,C) = A + C$

K-map for $E(A,B,C)$

\[
\begin{array}{cccc}
A & B & C & 00 & 01 & 11 & 10 \\
\hline
0 & 0 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 1 \\
\end{array}
\]

$E(A,B,C) = A + B$
K-map for \( F(A,B,C) \)

\[
\begin{array}{cccc}
 AB & 00 & 01 & 11 & 10 \\
 C & 0 & & & \\
 0 & 0 & 1 & X & X \\
 1 & 1 & X & X & X \\
\end{array}
\]

\( F(A,B,C) = B + C \)

Combinational Components

Levels of Integration:

- SSI (small scale integration)
  - 1-10 gates
- MSI (medium scale integration)
  - 10-100 gates
- LSI (large scale integration)
  - 100-10,000 gates
- VLSI (very large scale integration)
  - More than 10,000 gates
Combinational Components

Control:
- Multiplexer
- Demultiplexer
- Decoder
- Encoder

Data Manipulation:
- Magnitude Comparator
- Adder
- Shifter

Multiplexer

A multiplexer is used to select one input (out of several inputs) to be the circuit output

- $2^n$ data inputs
- n selection inputs
- 1 output

The n selection bits determine which of the $2^n$ data inputs is routed to the output
Ex: 4-to-1 Multiplexer

The 2-bit selection signal selects 1 of the 4 data inputs to become the output

Ex: 4-to-1 Multiplexer

Design of 2-to-1 Multiplexer

Characteristic table:
Karnaugh Map

\[ O = S'I_0 + SI_1 \]

Circuit Diagram:
Design of 4-to-1 Multiplexer

Truth table: 6 inputs, so 64 rows in truth table

Characteristic table:

<table>
<thead>
<tr>
<th>$S_1$</th>
<th>$S_0$</th>
<th>O</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>$I_0$</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>$I_1$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>$I_2$</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>$I_3$</td>
</tr>
</tbody>
</table>
Circuit diagram

Block Diagram